

ADA057870

②

LEVEL III

A057871-
A057873

RADC-TR-78-155, Volume I (of five)
Final Technical Report
July 1978



BAYESIAN SOFTWARE PREDICTION MODELS, *Volume I.*
An Imperfect Debugging Model for Reliability and
other Quantitative Measures of Software Systems

Amrit L. Goel
K. Okumoto

Syracuse University

AD No. _____
DC FILE COPY

Approved for public release; distribution unlimited.

DDC
RECEIVED
AUG 28 1978
B

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

78 08 22 060

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-78-155, Volume I (of five) has been reviewed and is approved for publication.

APPROVED:

Alan N. Sukert

ALAN N. SUKERT
Project Engineer

APPROVED:

Alan R. Barnum

ALAN R. BARNUM
Assistant Chief
Information Sciences Division

FOR THE COMMANDER:

John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (ISIS) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.

19 TR-78-155-VOL-1

14 TR-78-1

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
18 RADC-TR-78-155, Vol I (of five)		9
4. TITLE (and Subtitle)		5. TYPE OF REPORT & PERIOD COVERED
6 BAYESIAN SOFTWARE PREDICTION MODELS. Volume I. An Imperfect Debugging Model for Reliability and other Quantitative Measures of Software Systems.		Final Technical Report. Dec 75 - Mar 78
7. AUTHOR(S)		8. PERFORMING ORG. REPORT NUMBER
10 Amrit L./Goel K. Okumoto		Technical Report No. 78-1
9. PERFORMING ORGANIZATION NAME AND ADDRESS		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
Syracuse University Syracuse NY 13210		15 F30602-76-C-0097
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE
Rome Air Development Center (ISIS) Griffiss AFB NY 13441		11 Jul 78
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		13. NUMBER OF PAGES
Same		45
16. DISTRIBUTION STATEMENT (of this Report)		15. SECURITY CLASS. (of this report)
Approved for public release; distribution unlimited.		UNCLASSIFIED
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
Same		N/A
18. SUPPLEMENTARY NOTES		
RADC Project Engineer: Alan N. Sukert (ISIS)		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)		
Software Error Prediction Gamma Distributions Software Error Models Imperfect Debugging Markov Processes Stochastic Process		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)		
In this report a stochastic model for software failure phenomena is developed for the case when the errors are not corrected with certainty. Expressions for several quantities of interest are derived to establish quantitative measures for software performance assessment. Approximations for large-scale software systems using a gamma distribution are also discussed. Numerical examples are used to illustrate the computations and usefulness of various quantities. Volume V will be published at a later date.		

DDC
RECEIVED
AUG 23 1978
B

DD FORM 1 JAN 73 1473

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

339 60078 08 22 060

Lee

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

Block 7.

¹Professor, Department of Industrial Engineering & Operations Research,
and School of Computer and Information Science, Syracuse University.

²Research Assistant, Department of Industrial Engineering and
Operations Research.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

TABLE OF CONTENTS

1.	INTRODUCTION	1
2.	MODEL DEVELOPMENT.	2
3.	DERIVATION OF VARIOUS QUANTITIES OF INTEREST	9
3.1	Distribution of Time to a Completely Debugged Software System.	9
3.2	Distribution of Time to a Specified Number of Remaining Errors	13
3.3	Distribution of Number of Remaining Errors	18
3.4	Expected Number of Errors Detected by Time t	23
4.	DISTRIBUTION OF TIME BETWEEN SOFTWARE FAILURES	27
5.	GAMMA APPROXIMATION FOR A LARGE-SCALE SOFTWARE SYSTEM.	33
6.	CONCLUDING REMARKS	41
7.	SELECTED REFERENCES.	42

ACCESSION FOR		
NTIS	NTIS Section	<input checked="" type="checkbox"/>
DOC	DOC Section	<input type="checkbox"/>
UNANNOUNCED		<input type="checkbox"/>
NOTIFICATION		
BY		
INSTITUTION/AVAILABILITY CODES		
Doc.	AVAIL.	and/or SPECIM.
A		

LIST OF FIGURES

Figure		Page
2.1	A Diagrammatic Representation of Transitions Between States of $X(t)$	4
2.2	A Typical Realization of the $X(t)$ Process.	6
3.1	CDF of Time to Completely Debugged Software.	12
3.2	PDF of Time to a Specified Number (n_0) of Remaining Errors	15
3.3	CDF of Time to a Specified Number (n_0) of Remaining Errors	16
3.4	Probability Distributions for Number of Remaining Errors at Time t	20
3.5	Expected Number of Remaining Errors versus Time t . . .	21
3.6	Expected Number of Errors Detected by Time t	24
3.7	Plots of $M_N(t)$ and $D_N(t)$	26
4.1	Reliability Growth Curves.	30
4.2	PDF of Time Between Software Failures.	31
4.3	Failure Rates of Time Between Software Failures. . .	32
5.1	Relative Loss for the Third and Fourth Moments . . .	38
5.2	Plots of First Passage Times Using Gamma Approxima- tion	39
5.3	State Occupancy Probabilities Using Gamma Approximations	40

LIST OF TABLES

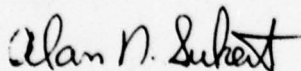
Table	Page
3.1 Mean and Variance of First Passage Time for Various n_0	17
5.1 Gamma Approximation for First Passage Time Distributions.	36

EVALUATION

The necessity for more complex software systems in such areas as command and control and avionics has led to the desire for better methods for predicting software errors to insure that software produced is of higher quality and of lower cost. This desire has been expressed in numerous industry and Government sponsored conferences, as well as in documents such as the Joint Commanders' Software Reliability Working Group Report (Nov 1975). As a result, numerous efforts have been initiated to develop and validate mathematical models for predicting such quantities as the number of remaining errors in a software package, the time to achieve a desired reliability level, and a measure of the software reliability. However, early efforts have not produced models with the desired accuracy of prediction and with the necessary confidence limits for general model usage.

This effort was initiated in response to this need for developing better and more accurate software error prediction models and fits into the goals of RADC TPO No. 5, Software Cost Reduction (formerly RADC TPO No. 11, Software Sciences Technology), in the subthrust of Software Quality (Software Modeling). This report summarizes the development of a mathematical model for predicting quantities such as the expected number of remaining errors, achieved reliability, and time to detect and correct a specified number of errors that assumes a software error is not corrected at a given time with probability 1 (i.e. imperfect debugging). The importance of this development is that it represents the first attempt to develop software error prediction models that incorporate imperfect debugging, and thus more closely reflect the actual software error detection and correction process.

The theory and equations developed under this effort will lead to much needed predictive measures for use by software managers in more accurately tracking software development projects in terms of test time needed to achieve given reliability and error objectives. In addition, the associated confidence limits and other related statistical quantities developed under this effort will insure more widespread use of these modeling techniques. Finally, the predictive measures and equations developed under this effort will be applicable to current Air Force software development projects and thus help to produce the high quality, low cost software needed for today's systems.



ALAN N. SUKERT
Project Engineer

1. INTRODUCTION

A considerable emphasis has been placed in recent years on the study of software error phenomena with the objective of developing analytical models which can be used to obtain quantitative measures for software performance. Most of these studies assume an exponential distribution for times between software errors with a failure rate that depends on the number of remaining errors, see for example, [3, 6, 8, 9, 10, 11, 13, 15, 18].

A key assumption made in most of these studies is that the errors are removed with certainty, when detected. However, as pointed out in Miyamoto [7] and Thayer et al. [15], in practice errors are not always corrected when detected. The existing models do not provide a solution for such situations. The purpose of this report, then, is to develop an analytical model for software error phenomenon when the errors are not removed/corrected with certainty, i.e., for the case of imperfect debugging. The model is developed in Section 2 and expressions for the following quantities of interest are derived in Section 3:

- (i) Distribution of time to a completely debugged system.
- (ii) Distribution of time to a specified number of remaining errors.
- (iii) Distribution of number of remaining errors.
- (iv) Expected number of errors detected by time t .

The distribution of time between software failure is obtained in Section 4 and approximate solutions using a gamma distribution are discussed in Section 5. Numerical examples are used to illustrate the computations and usefulness of various quantities.

2. MODEL DEVELOPMENT

The following assumptions are made for developing the model.

- (i) The error causing a software failure, when detected, is corrected with probability $p(0 \leq p \leq 1)$, while with probability $q(p+q=1)$ we fail to completely remove it. Thus, q is the probability of imperfect debugging.
- (ii) Errors in the software package are independent of each other and have a constant occurrence rate λ .
- (iii) The probability of two or more errors occurring simultaneously is negligible.
- (iv) The time to remove an error is considered to be negligible in this model.
- (v) No new errors are introduced during the debugging process.
- (vi) At most one error is removed at correction time.

Let $X(t)$ denote the number of errors remaining in the package at time t . We will use this random variable to describe the state of the error process at time t . Further, let N be the number of errors at the beginning of the debugging phase, i.e., $X(0) = N$.

Suppose that there are i errors in the package at some time. Then from assumption (i), we note that after the occurrence of the next failure

$$X(t) = \begin{cases} i-1 & \text{with probability } p \\ i & \text{with probability } q \end{cases} \quad (2.1)$$

In other words, if we were to observe the $X(t)$ process at times of software failures, then its behavior is governed by equation (2.1). The transition probabilities P_{ij} from state i to state j ,

$i, j = 0, 1, 2, \dots, N$, are given by

$$P = [P_{ij}] = \begin{matrix} & \begin{matrix} 0 & 1 & 2 \dots i-2 & i-1 & i & \dots & N-2 & N-1 & N \end{matrix} \\ \begin{matrix} 0 \\ 1 \\ 2 \\ \vdots \\ i-1 \\ i \\ \vdots \\ N-1 \\ N \end{matrix} & \begin{bmatrix} 1 & 0 & 0 & 0 & - & - & - & - & 0 & 0 \\ p & q & 0 & 0 & - & - & - & - & 0 & 0 \\ 0 & p & q & 0 & & & & & 0 & 0 \\ \vdots & & \ddots & \ddots & & & & & \vdots & \vdots \\ - & - & - & p & q & & & & \vdots & \vdots \\ - & - & - & - & p & q & & & \vdots & \vdots \\ 0 & 0 & - & - & - & - & p & q & 0 \\ 0 & 0 & - & - & - & - & - & 0 & p & q \end{bmatrix} \end{matrix} \quad (2.2)$$

A diagrammatic representation of transitions between states corresponding to equation (2.2) is given in Figure 2.1.

Now, assumptions (i) and (ii) imply that the times between successive software failures (error occurrences) follow an exponential distribution. Suppose at some time $t = \tau$, $x(\tau) = i$, $i = 0, 1, \dots, N$. Then the probability density function (pdf) $f_i(t)$ of the time to next failure, T_i , is given by the distribution of the first order statistic of i exponential distributions each with parameter λ , i.e.,

$$f_i(t) = \binom{i}{1} \lambda e^{-\lambda t} \cdot (e^{-\lambda t})^{i-1}$$

$$\text{or} \quad f_i(t) = i\lambda \cdot e^{-i\lambda t} \quad (2.3)$$

and the cumulative distribution function (cdf) is given by

$$F_i(t) = 1 - e^{-i\lambda t}. \quad (2.4)$$

We note that even though the stochastic process $X(t)$ makes transitions from state to state in accordance with equation (2.2), the times spent in various states are random and are given by equation (2.3). Hence $\{X(t), t \geq 0\}$ forms a semi-Markov process. A typical realization of this process is shown in Figure 2.2. It should be pointed out that in our formulation the process $X(t)$ undergoes both real and virtual transitions. This means that after an attempt to remove an error the state of $X(t)$ may change or may remain unchanged. In Figure 2.2 real transitions occur at states $N, N-2$ and i while a virtual transition occurs at state $N-1$.

Let $Q_{ij}(t)$ denote the one step transition probability that after making a transition into state i , the process $X(t)$ next makes a transition into state j by time t . In other words if a software package has i remaining errors at time zero, then $Q_{ij}(t)$ represents the probability that the next failure, resulting in j remaining errors, will be by time t . Hence, for $i, j = 0, 1, 2, \dots, N$, we can write

$$Q_{ij}(t) = \int_0^t P\{X(u) = j, T_i = u | X(0) = i\} \cdot du.$$

Since the events $\{X(u) = j\}$ and $\{T_i = u\}$ are independent, we get

$$\begin{aligned} Q_{ij}(t) &= \int_0^t P\{X(u) = j | X(0) = i\} P\{T_i = u | X(0) = i\} \cdot du \\ &= \int_0^t P_{ij} \cdot P\{T_i = u | X(0) = i\} \cdot du \\ &= P_{ij} \int_0^t \lambda \cdot e^{-\lambda u} \cdot du \\ &= P_{ij} \cdot (1 - e^{-\lambda t}) \end{aligned}$$

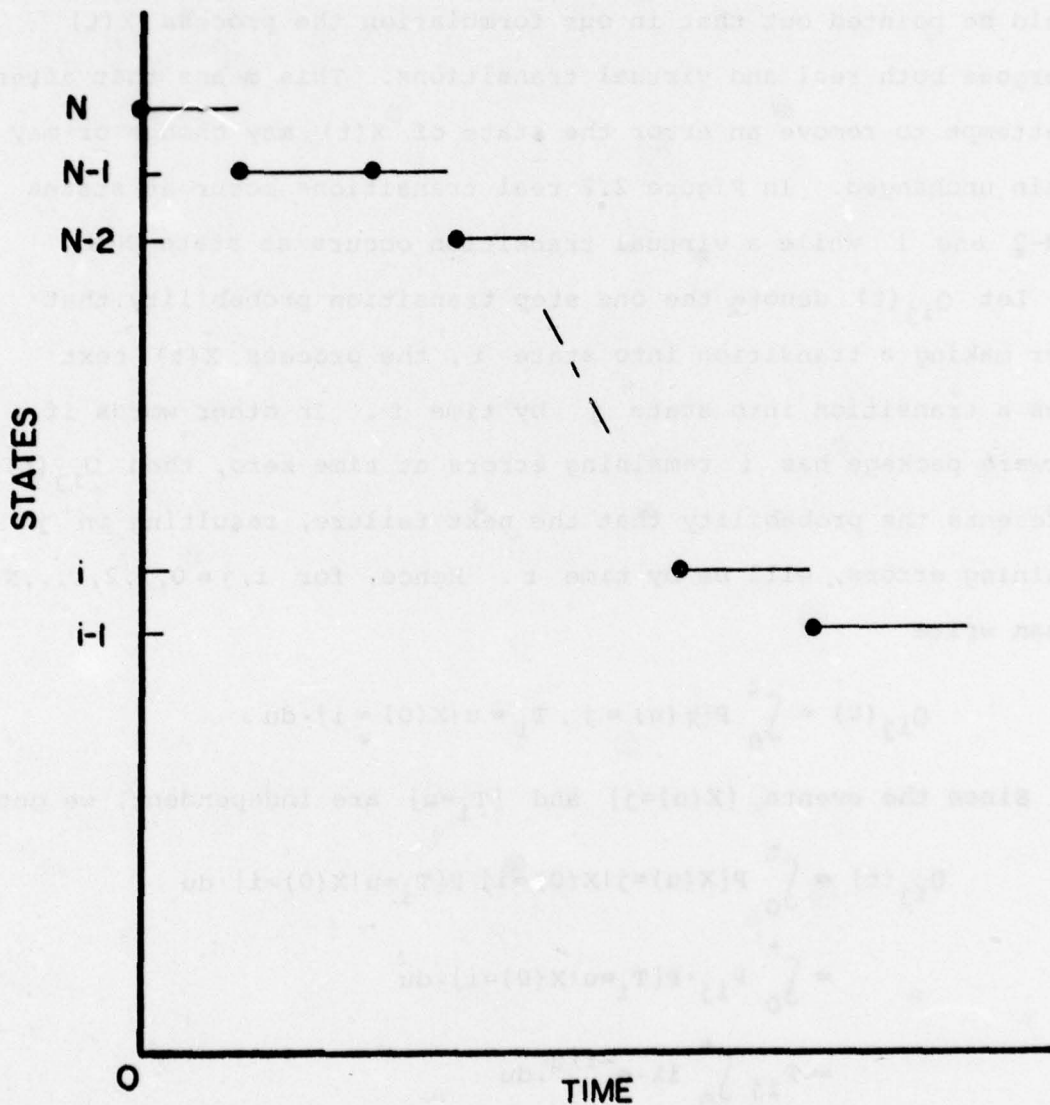


Figure 2.2 A Typical Realization of the $X(t)$ Process

$$\text{or } Q_{ij}(t) = P_{ij} \cdot F_i(t) \quad (2.5)$$

for $i, j = 0, 1, 2, \dots, N$.

It is obvious that $Q_{ij}(t)$ must satisfy

$$Q_{ij}(t) \geq 0, \quad i, j = 0, 1, 2, \dots, N, \quad t \geq 0$$

and

$$\sum_{j=0}^N Q_{ij}(\infty) = p+q = 1, \quad i = 0, 1, \dots, N.$$

The probabilities $Q_{ij}(t)$ are obtained by multiplying the probabilities P_{ij} from (2.2) and $F_i(t)$ from (2.4). Thus, for example,

$$Q_{N,N-1}(t) = P_{N,N-1} \cdot F_N(t)$$

$$\text{or } Q_{N,N-1}(t) = p(1 - e^{-N\lambda t}).$$

Proceeding similarly for all i, j we get $\{Q_{ij}(t)\}$ as shown in Equation (2.6) on the following page.

For known parameters N , p and λ , the probabilities $Q_{ij}(t)$ are obtained from Equation (2.6). This equation represents the basic model that will be used in the following sections for obtaining the various quantities of interest for the software error phenomenon.

(2.6)

$$\{Q_{ij}(t)\} = \begin{bmatrix} 0 & 1 & pF_1(t) & qF_1(t) & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 2 & pF_1(t) & 0 & 0 & 0 & \dots & 0 & 0 \\ \vdots & qF_1(t) & pF_2(t) & qF_2(t) & \vdots & \ddots & \vdots & \vdots \\ N-1 & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ N & 0 & 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix}$$

3. DERIVATION OF VARIOUS QUANTITIES OF INTEREST

3.1 Distribution of Time to a Completely Debugged Software System

Suppose i is the number of errors remaining in a software system at some time during the debugging process. Let $g_{i,0}(t)$ and $G_{i,0}(t)$ denote the pdf and cdf, respectively, of the first passage time from i to 0. In other words $g_{i,0}(t)$ and $G_{i,0}(t)$ represent, respectively, the pdf and cdf of the time required to obtain a completely debugged software system when the initial number of errors is i .

Recall that at time zero, $X(0) = N$ and at the time of the next failure

$$X(t) = \begin{cases} N-1 & \text{with probability } p \\ N & \text{with probability } q \end{cases} \quad (3.1)$$

as shown in Figure 2.1. Now, from the definition of $Q_{i,j}(t)$, the probability of going from N to $N-1$ errors in time $[u, u+du]$ is $dQ_{N,N-1}(u)$. Then the process $X(t)$ restarts with $(N-1)$ remaining errors at time u and the cdf of the first passage time is $G_{N-1,0}(t-u)$. For the case of perfect debugging the cdf of the first passage time is

$$\int_0^t G_{N-1,0}(t-u) \cdot dQ_{N,N-1}(u) = Q_{N,N-1} * G_{N-1,0}(t), \quad (3.2)$$

where $*$ denotes convolution.

Similarly, if the debugging at the first error occurrence is imperfect, the cdf of the first passage time is

$$\int_0^t G_{N,0}(t-u) \cdot dQ_{N,N}(u) = Q_{N,N} * G_{N,0}(t) \quad (3.3)$$

Since the events depicted in Equations (3.2) and (3.3) are mutually exclusive, we get the renewal equation

$$G_{N,0}(t) = Q_{N,N-1} * G_{N-1,0}(t) + Q_{N,N} * G_{N,0}(t) \quad (3.4)$$

In general, we get the renewal equation

$$G_{i,0}(t) = Q_{i,i-1} * G_{i-1,0}(t) + Q_{i,i} * G_{i,0}(t) \quad (3.5)$$

for $i = 1, 2, \dots, N$

where $G_{0,0}(t) = 1$.

We use Laplace-Stieltjes (L-S) transforms to solve renewal equations (3.5), where the L-S of $G_{i,0}(t)$ is defined as:

$$\tilde{G}_{i,0}(s) = \int_0^\infty e^{-st} \cdot dG_{i,0}(t) \quad (3.6)$$

From (3.5) we get

$$\tilde{G}_{i,0}(s) = \tilde{Q}_{i,i-1}(s) \tilde{G}_{i-1,0}(s) + \tilde{Q}_{i,i}(s) \tilde{G}_{i,0}(s), \quad i = 1, 2, \dots, N \quad (3.7)$$

where

$$\tilde{Q}_{i,i-1}(s) = \frac{i p \lambda}{s + i \lambda}, \quad (3.8)$$

and

$$\tilde{Q}_{i,i}(s) = \frac{i q \lambda}{s + i \lambda}. \quad (3.9)$$

Solving (3.7) recursively, we get the L-S transform of $G_{N,0}(t)$ as

$$\tilde{G}_{N,0}(s) = \prod_{j=1}^N \frac{j p \lambda}{s + j p \lambda} = \sum_{j=1}^N C_{N,j} \frac{j p \lambda}{s + j p \lambda} \quad (3.10)$$

where

$$C_{N,j} = \binom{N}{j} (-1)^{j-1} \quad (3.11)$$

By taking the inverse of $\tilde{G}_{N,0}(s)$, the cdf of the first passage time from N to 0 is:

$$G_{N,0}(t) = \sum_{j=1}^N C_{N,j} (1 - e^{-j p \lambda t}). \quad (3.12)$$

The pdf of the first passage time from N to 0 is given by

$$g_{N,0}(t) = \sum_{j=1}^N C_{N,j} \cdot j p \lambda \cdot e^{-j p \lambda t}. \quad (3.13)$$

To illustrate the above result let us consider a software system with $N=10$, $\lambda=0.02$ and $p=0.8$. Then

$$G_{10,0}(t) = \sum_{j=1}^{10} \binom{10}{j} (-1)^{j-1} \cdot [1 - e^{-j(.8)(.02)t}].$$

The values of this function for various t are plotted in Figure 3.1. From this plot we note that the probability of getting an error free system by 275 time units is 0.9 and by 500 units is 1.0. Such a plot is useful for calculating the time required to get an error free system with a desired probability.

Similar plots for values of $p=.85$, $.90$, $.95$ and 1.0 are also shown in Figure 3.1. As would be expected the cdf for a larger p dominates that for a smaller p . In other words the better the debugger, the faster is the process of debugging.

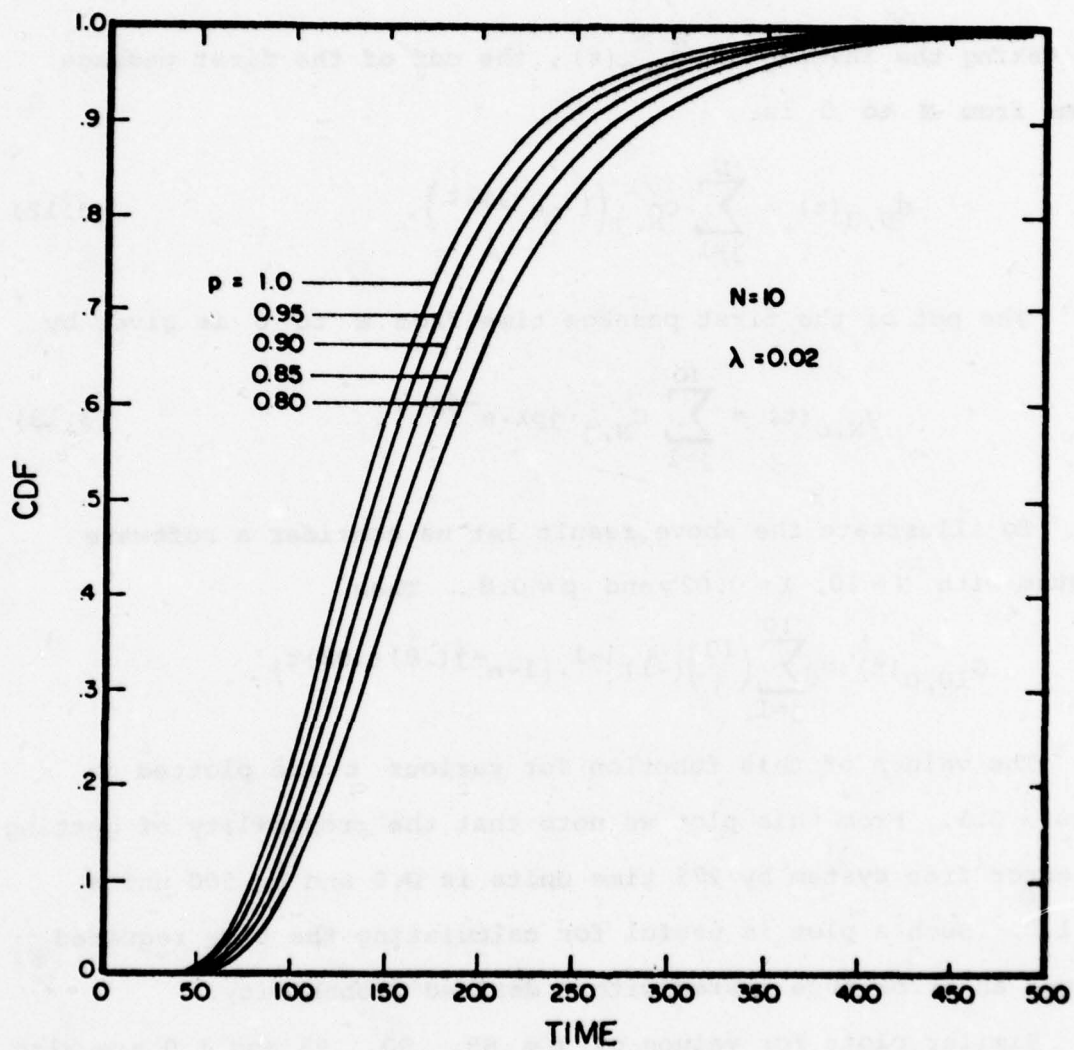


Figure 3.1 CDF of Time to a Completely Debugged Software System

3.2 Distribution of Time to a Specified Number of Remaining Errors

In many instances a completely debugged software is not cost effective and we may be willing to tolerate a certain number of remaining errors, say n_0 , which will ensure some desired reliability. The distribution of time to n_0 is then of interest.

Using an approach similar to that of Section 3.1 we get the renewal equation

$$G_{i,n_0}(t) = Q_{i,i-1} * G_{i-1,n_0}(t) + Q_{i,i} * G_{i,n_0}(t),$$

$$\text{for } i = n_0 + 1, \dots, N \quad (3.14)$$

where $G_{n_0,n_0}(t) = 1$.

Then the L-S transform of $G_{N,n_0}(t)$ is given by

$$\tilde{G}_{N,n_0}(s) = \prod_{j=n_0+1}^N \frac{j p \lambda}{s + j p \lambda} = \sum_{j=1}^{n-n_0} B_{N,j,n_0} \frac{(n_0+j) p \lambda}{s + (n_0+j) p \lambda} \quad (3.15)$$

where

$$B_{N,j,n_0} = \frac{N!}{n_0! j! (N-n_0-j)!} (-1)^{j-1} \frac{j}{n_0+j}. \quad (3.16)$$

The cdf is obtained by taking the inverse L-S transform of

$\tilde{G}_{N,n_0}(s)$,

$$G_{N,n_0}(t) = \sum_{j=1}^{N-n_0} B_{N,j,n_0} \{1 - e^{-(n_0+j) p \lambda t}\}, \quad (3.17)$$

and the pdf is

$$g_{N,n_0}(t) = \sum_{j=1}^{N-n_0} B_{N,j,n_0} (n_0+j) p \lambda e^{-(n_0+j) p \lambda t} \quad (3.18)$$

To see the nature of the pdf and cdf, let us consider the case when $N=10$, $\lambda = .02$ and $p=0.9$. These are shown in Figures 3.2 and 3.3, respectively, for various values of n_0 . The plots are self explanatory.

Now let a random variable T_{N,n_0} denote the first passage time from N to n_0 errors. Then, from (3.18) we can obtain the l^{th} moment of T_{N,n_0} as

$$E[T_{N,n_0}^l] = \sum_{j=1}^{N-n_0} B_{N,j,n_0} \frac{\Gamma(l+1)}{[(n_0+j)jp]^l} . \quad (3.19)$$

From (3.19), the mean and variance are

$$ET_{N,n_0} = \sum_{j=1}^{N-n_0} B_{N,j,n_0} / (n_0+j)p\lambda \quad (3.20)$$

$$\text{Var}(T_{N,n_0}) = ET_{N,n_0}^2 - (ET_{N,n_0})^2 \quad (3.21)$$

The values for mean and variance of first passage time for various n_0 are given in Table 3.1, where $N=10$, $p=0.9$ and $\lambda = 0.02$.

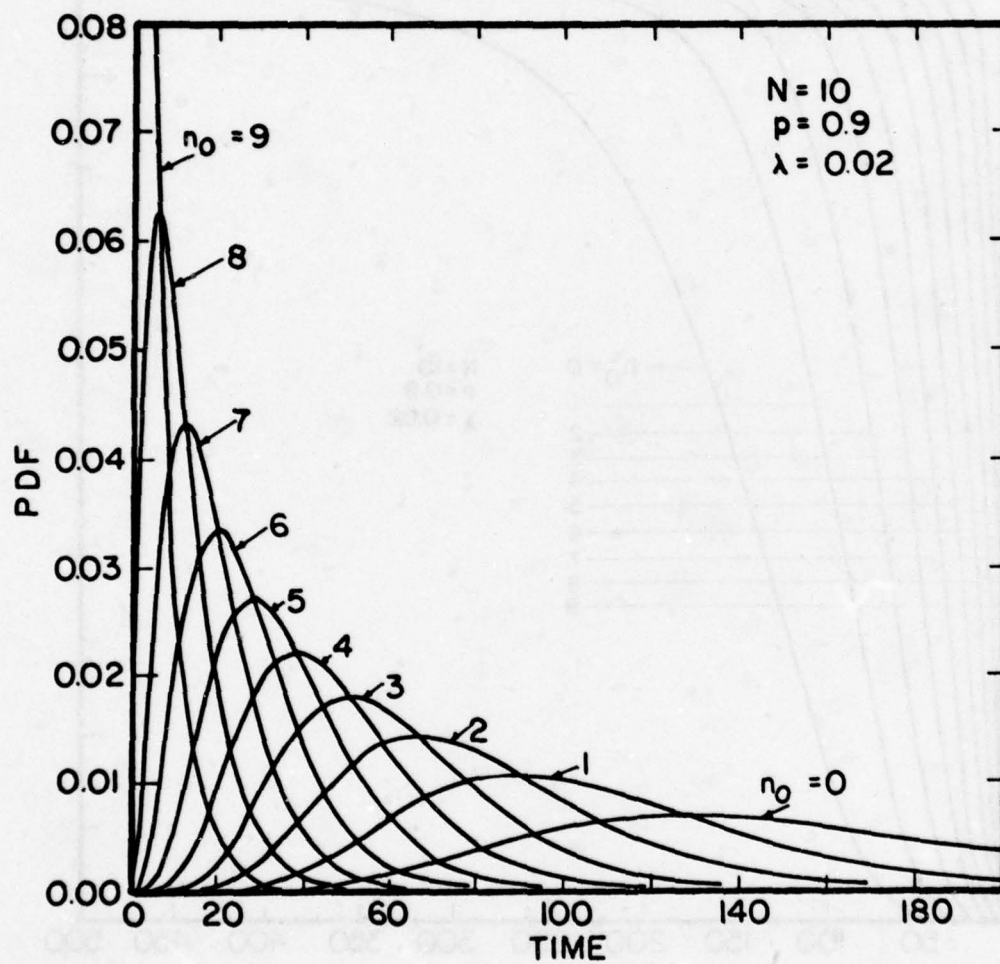


Figure 3.2 PDF of Time to a Specified Number (n_0) of Remaining Errors

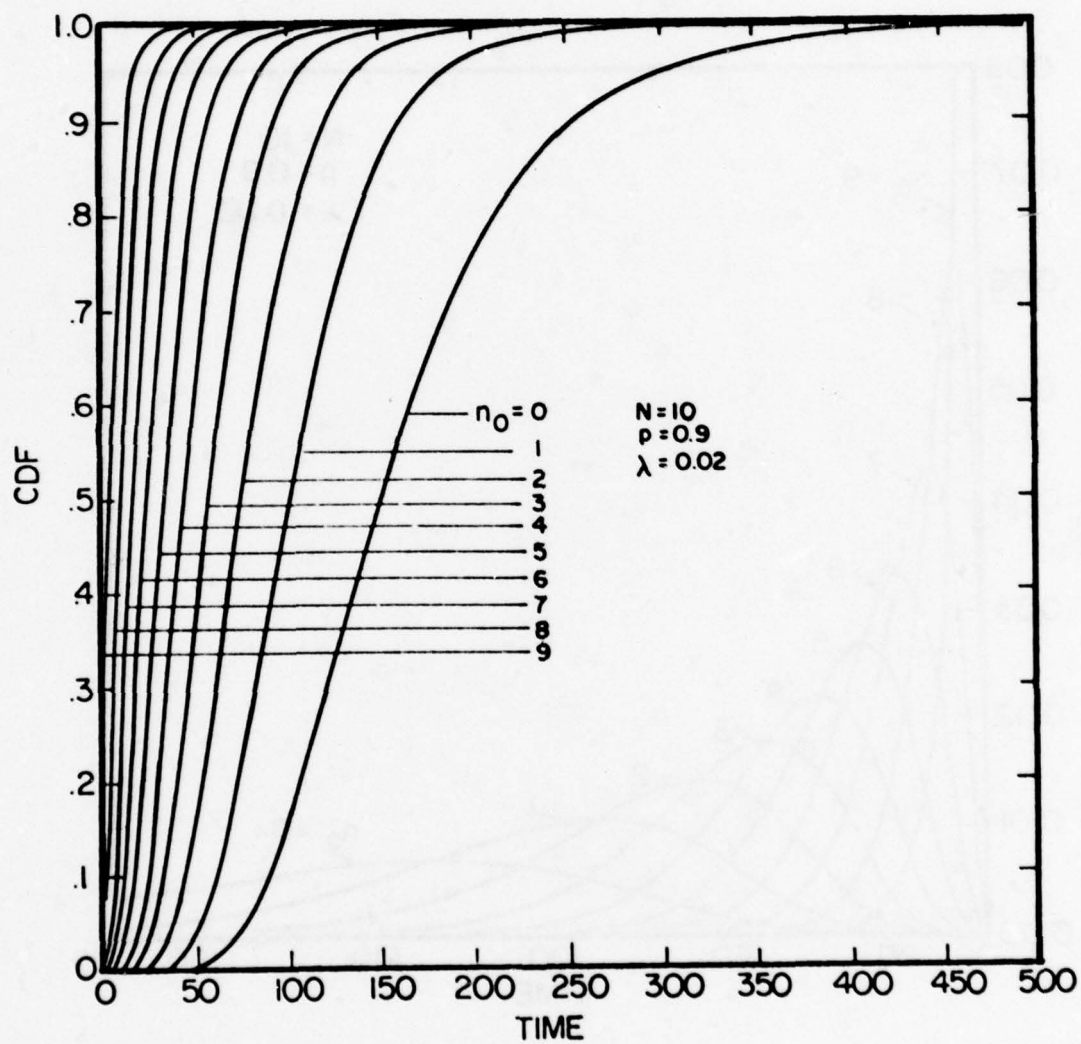


Figure 3.3 CDF of Time to a Specified Number (n_0) of Remaining Errors

Table 3.1

Mean and Variance of First Passage Time for Various n_0

($N = 10$, $p = 0.9$, $\lambda = 0.02$)

n_0	Mean	Variance	Standard Deviation
9	5.56	30.86	5.56
8	11.73	68.97	8.30
7	18.67	117.19	10.83
6	26.61	180.18	13.42
5	35.87	265.92	16.31
4	46.98	389.37	19.73
3	60.87	582.27	24.13
2	79.39	925.21	30.42
1	107.16	1696.81	41.19
0	162.72	4783.23	69.16

3.3 Distribution of Number of Remaining Errors

First, we develop the expressions for the distribution of the number of remaining errors after a specified time period, t . Then, the expected number of remaining errors at time t is obtained.

Let $P_{N,n_0}(t)$ represent the probability that there are n_0 errors remaining in a software package at time t , given that there are N errors at the beginning of debugging, i.e.,

$$P_{N,n_0}(t) = P\{X(t) = n_0 | X(0) = N\} \quad (3.22)$$

which is the so-called state occupancy probability. Conditioning on the next failure and following an approach similar to that of Section 3.1, we get the following renewal equation.

$$P_{n_0,n_0}(t) = e^{-n_0\lambda t} + Q_{n_0,n_0} * P_{n_0,n_0}(t), \quad n_0 \leq N. \quad (3.23)$$

Conditioning on the first passage time, we get

$$P_{N,n_0}(t) = P_{n_0,n_0} * G_{N,n_0}(t), \quad n_0 < N. \quad (3.24)$$

By taking the L-S transform of $P_{n_0,n_0}(t)$ and rearranging, we get

$$\tilde{P}_{n_0,n_0}(s) = \frac{s}{s + n_0 p \lambda} = 1 - \frac{n_0 p \lambda}{s + n_0 p \lambda} \quad (3.25)$$

Substituting (3.25) into the L-S transform of $P_{N,n_0}(t)$, we obtain

$$\begin{aligned} \tilde{P}_{N,n_0}(s) &= \tilde{G}_{N,n_0}(s) - \frac{n_0 p \lambda}{s + n_0 p \lambda} \tilde{G}_{N,n_0}(s) \\ &= \tilde{G}_{N,n_0}(s) - \tilde{G}_{N,n_0-1}(s). \end{aligned} \quad (3.26)$$

By taking the inverse L-S transform of $\tilde{P}_{N,n_0}(s)$ we get

$$P_{N,n_0}(t) = G_{N,n_0}(t) - G_{N,n_0-1}(t), \quad n_0 = 0, 1, 2, \dots, N \quad (3.27)$$

where

$$G_{N,N}(t) = 1,$$

$$G_{N,-1}(t) = 0.$$

Figure 3.4 shows $P_{N,n_0}(t)$ for various n_0 , where $N=10$, $p=0.9$, and $\lambda=0.02$. From this figure we can see how the distribution of the number of remaining errors changes with time.

Now, we obtain the expected number of remaining errors in the software at time t as follows:

$$\begin{aligned} E[X(t) | X(0)=N] &= \sum_{n_0=0}^N n_0 P_{N,n_0}(t) \\ &= \sum_{n_0=0}^N n_0 \{G_{N,n_0}(t) - G_{N,n_0-1}(t)\} \\ &= \sum_{n_0=0}^N \{1 - G_{N,n_0}(t)\} \end{aligned}$$

Now, using the expression in (3.17), we get

$$E[X(t) | X(0)=N] = Ne^{-p\lambda t}. \quad (3.28)$$

Figure 3.5 shows the expected number of remaining errors at time t for various p , where $N=10$, and $\lambda=0.02$. As can be seen, software errors can be eliminated faster if larger values of p are chosen. In other words, a good debugger can eliminate software errors fast. For example, for $n_0=1$ a debugger with $p=1$ requires

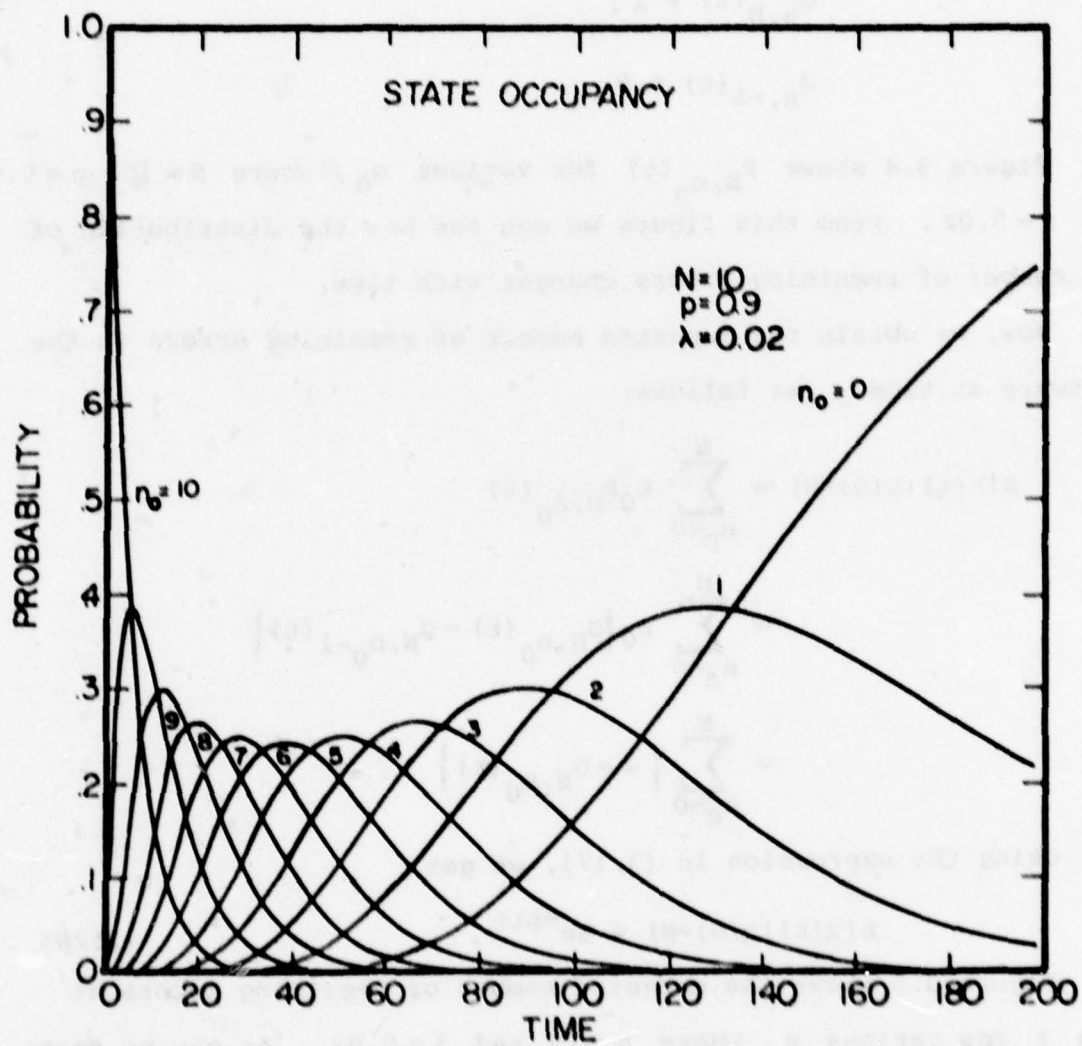


Figure 3.4 Probability Distributions of Number of Remaining Errors, n_0 , at Time t

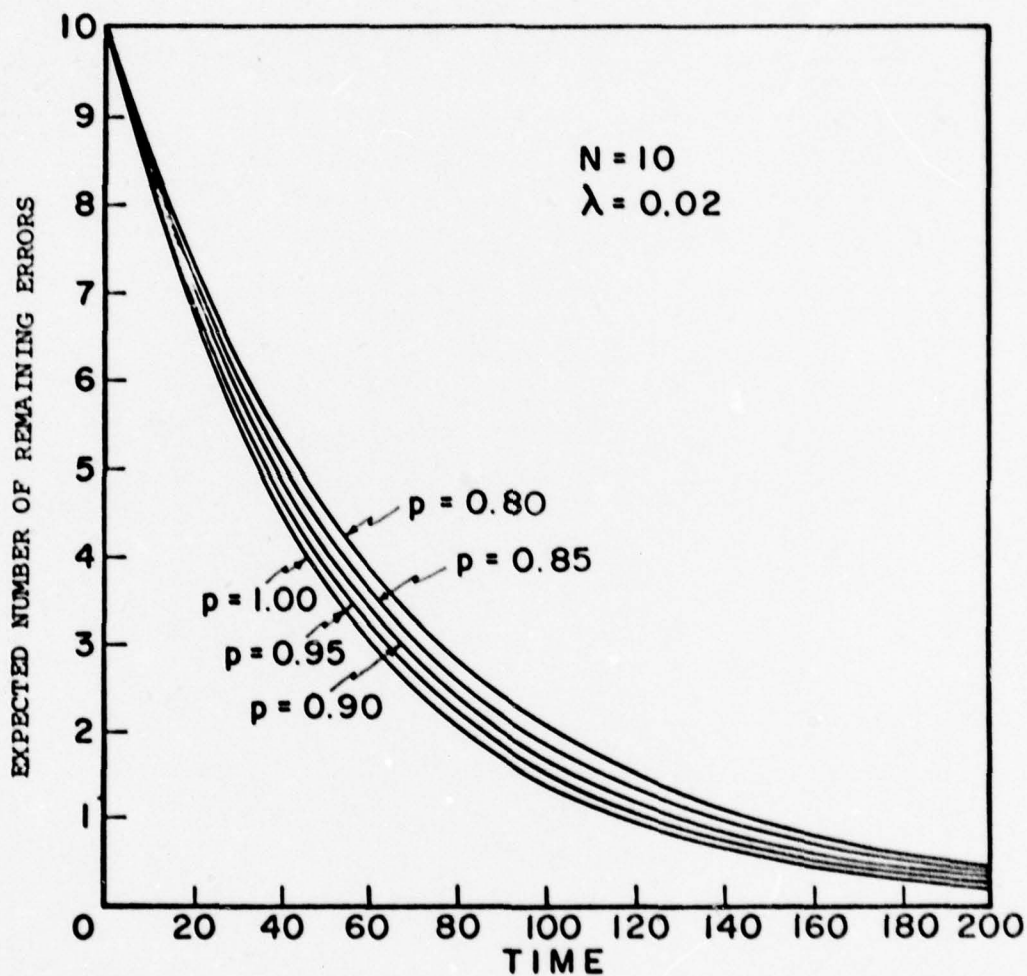
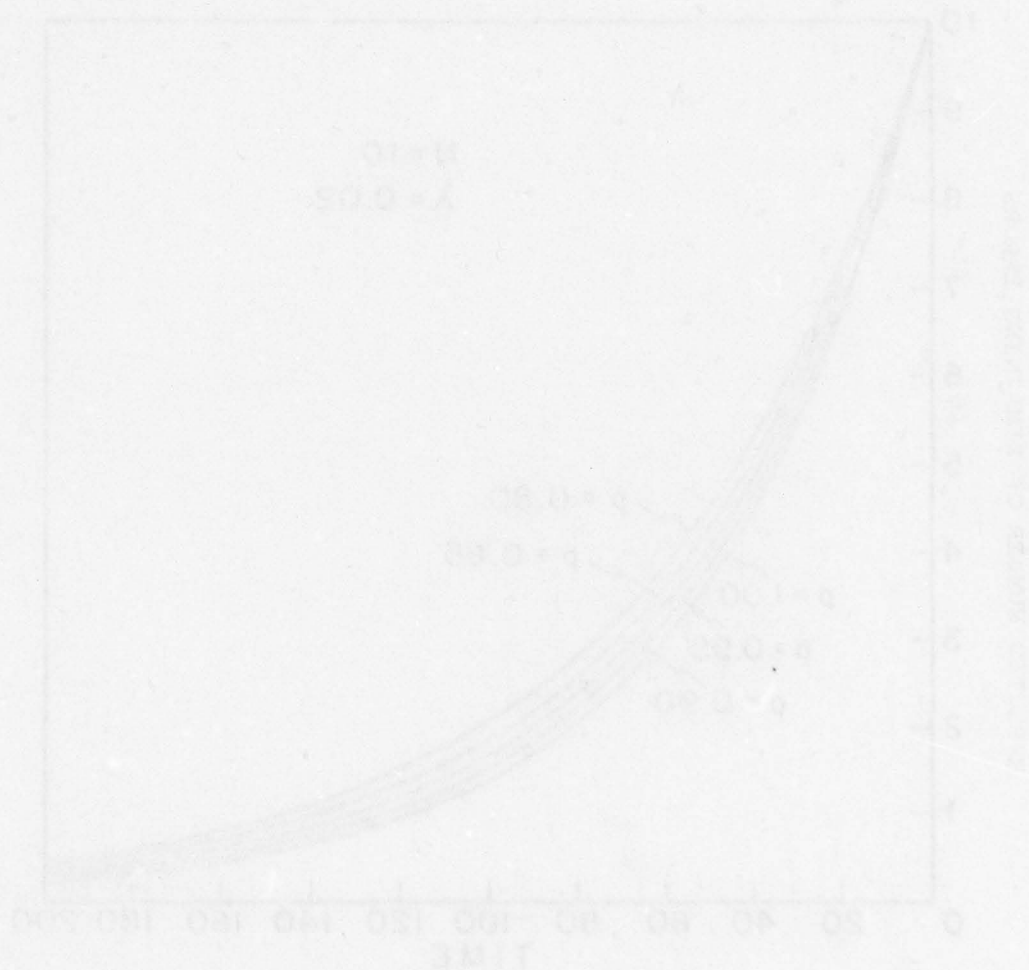


Figure 3.5 Expected Number of Remaining Errors versus Time t

debugging time $t = 118$, and the debugger with $p = 0.8$ requires $t = 148$. The difference between the two debuggers is 30 in the sense of expectation.



3.4 Expected Number of Errors Detected by Time t

We introduce a new random variable $N(t)$ which denotes the total number of errors detected by time t . The process $\{N(t), t \geq 0\}$ is called a counting process. We are interested in obtaining the expression for the expected number of errors detected, $M_N(t)$, during the debugging period, t , when the initial number of errors is N , i.e.

$$M_N(t) = E[N(t) | X(0) = N] \quad (3.29)$$

which is called a Markov renewal function. By conditioning on the next software failure, we obtain the renewal equations.

$$M_j(t) = F_j(t) + pM_{j-1} * F_j(t) + qM_j * F_j(t), \quad j = 1, 2, \dots, N \quad (3.30)$$

where $M_0(t) = 0$.

Using the L-S transforms of $M_j(t)$, $j=1, 2, \dots, N$, we get

$$\tilde{M}_N(s) = \frac{1}{p} \sum_{k=1}^N \prod_{j=k}^N \frac{jp\lambda}{s + jp\lambda} = \frac{1}{p} \sum_{k=1}^N \tilde{G}_{N,k-1}(s). \quad (3.31)$$

The expression for $M_N(t)$ in terms of the first passage time distribution is then given by

$$M_N(t) = \frac{1}{p} \sum_{k=1}^N G_{N,k-1}(t) = \frac{N}{p} (1 - e^{-p\lambda t}). \quad (3.32)$$

Note that if we let $t \rightarrow \infty$ we have

$$M_N(\infty) = \frac{N}{p} \quad (3.33)$$

which is the expected number of software errors detected by the end of debugging.

Figure 3.6 shows the expected number of errors detected by time t , $M_N(t)$, for various N when $p=0.9$ and $\lambda=0.02$.

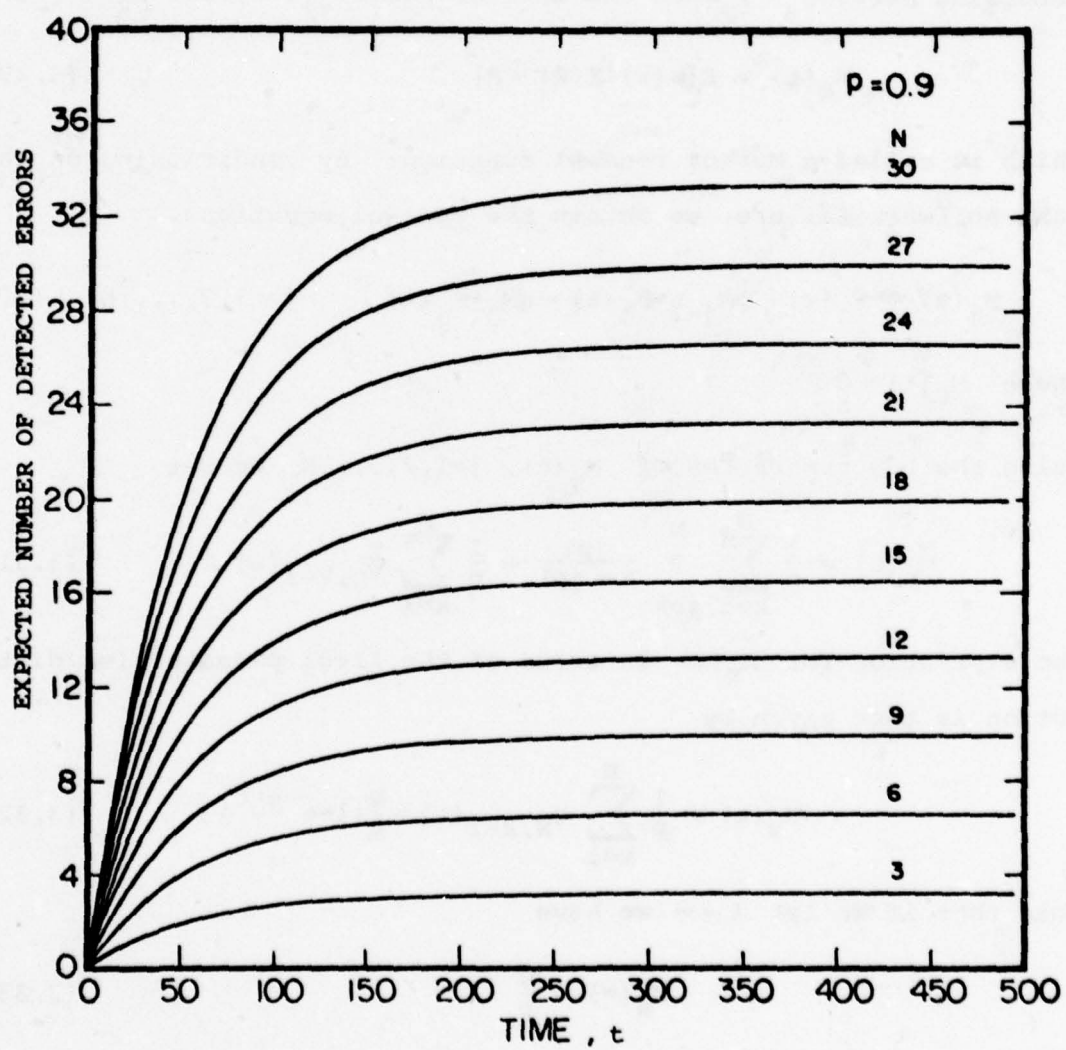


Figure 3.6 Expected Number of Errors
Detected by Time t for Various N

Let us now consider the case when the detected errors are separated as new errors and errors which were not corrected due to imperfect debugging. Let $N_I(t)$ be a random variable which denotes the total number of imperfect debugging errors by time t . Then we can show that

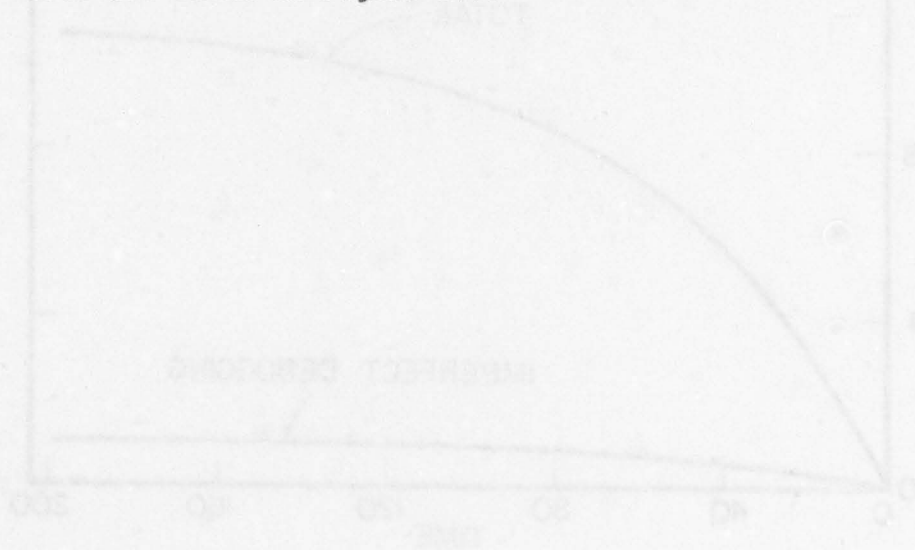
$$D_N(t) = qM_N(t) , \quad (3.34)$$

where

$$D_N(t) = E[N_I(t) | X(0)=N] .$$

Note that $D_N(\infty) = q \frac{N}{p}$.

Plots of $M_N(t)$ and $D_N(t)$ for the case when $N=10$, $p=0.9$ and $\lambda=0.02$ are shown in Figure 3.7.



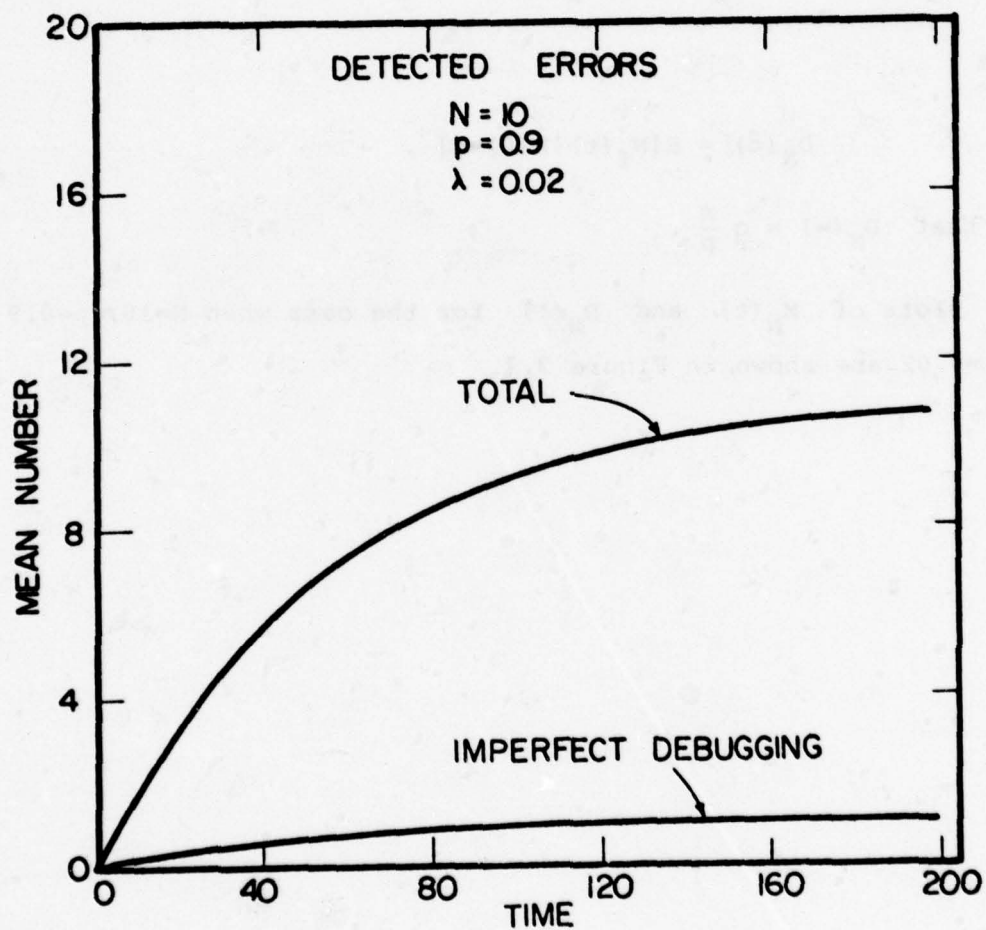


Figure 3.7 Plots of $M_N(t)$ and $D_N(t)$

4. DISTRIBUTION OF TIME BETWEEN SOFTWARE FAILURES

In the previous section we studied the stochastic behavior of the number of errors in the software system during the debugging period. In this section we investigate the distribution of the time between software failures and study the problem of reliability growth. From Section 2 recall that the random variable T_i denotes the time to next failure when the number of remaining errors is i and $F_i(t)$ is the cdf of T_i . Let X_k denote the time between the $(k-1)$ st and k th software failures and $\Phi_k(x)$ be the cdf of X_k . Note that X_k does depend on the number of remaining errors at the $(k-1)$ st failure but this number is not explicitly known. Further, let η_k , a r.v., denote the number of remaining errors between the $(k-1)$ st and k th software failures. Then, from Section 2 we have

$$\eta_1 = N \quad (4.1)$$

$$\Phi_1(x) = F_N(x), \quad (4.2)$$

and

$$\Phi_2(x) = pF_{N-1}(x) + qF_N(x). \quad (4.3)$$

In general, we have

$$\Phi_k(x) = P(X_k \leq x) = \sum_{i=N-(k-1)}^N P(X_k \leq x | \eta_k = i) P(\eta_k = i) \quad (4.4)$$

or

$$\begin{aligned}
\phi_k(x) &= \sum_{j=0}^{k-1} p(X_k \leq x | \eta_k = N-k+j+1) p(\eta_k = N-k+j+1) \\
&= \sum_{j=0}^{k-1} \binom{k-1}{j} p^{k-j-1} \cdot q^j F_{N-(k-j-1)}(x) .
\end{aligned} \tag{4.5}$$

This is called a mixture of exponential distributions with binomial mixing portions. As proved in Barlow and Proschan [1], $\phi_k(x)$ is a decreasing failure rate (DFR) distribution. The reliability function at the k th stage, i.e., between $(k-1)$ st and k th failure, is given by

$$\begin{aligned}
R_k(x) &= P\{X_k > x\} \\
&= 1 - \phi_k(x) \\
&= \sum_{j=0}^{k-1} \binom{k-1}{j} p^{k-j-1} q^j \bar{F}_{N-(k-j-1)}(x)
\end{aligned} \tag{4.6}$$

where

$$\bar{F}_N(x) = 1 - F_N(x) = e^{-N\lambda x} . \tag{4.7}$$

Also the corresponding failure rate is given by

$$r_k(x) = \phi_k(x) / R_k(x) , \tag{4.8}$$

where $\phi_k(x)$ is the p.d.f. of X_k . The behavior of $R_k(x)$ with respect to k is of interest. To study this behavior we have the following theorem.

Theorem: The reliability function $R_k(x)$ is increasing in k for any time $x > 0$, i.e.

$$R_k(x) < R_{k+1}(x), \quad k = 1, 2, \dots \quad (4.9)$$

Proof: It suffices to show that

$$\forall R_k(x) = R_{k+1}(x) - R_k(x) \quad (4.10)$$

is positive for $x > 0$. Then we have

$$\forall R_k(x) = \sum_{j=0}^{k-1} \binom{k-1}{j} p^{k-j} q^j \{ \bar{F}_{N-(k-j)}(x) - \bar{F}_{N-(k-j-1)}(x) \}. \quad (4.11)$$

It holds that for $x > 0$, $j = 0, 1, 2, \dots$

$$\bar{F}_{N-(k-1)}(x) > \bar{F}_{N-(k-j-1)}(x). \quad (4.12)$$

Hence we get

$$\forall R_k(x) > 0 \quad \text{for } x > 0. \quad (4.13)$$

Q.E.D.

The reliability growth curves are shown in Figure 4.1, where $N = 10$, $p = 0.9$ and $\lambda = 0.02$. The p.d.f.'s and the failure rates of X_k are shown in Figures 4.2 and 4.3, respectively.

Note that the number of software errors remaining at the time between $(k-1)$ st and k th software failures is $N-(k-I-1)$, where the random variable I is distributed as a binomial with $(k-1, q)$. Therefore, the expected number of software errors remaining is given by $N-p(k-1)$. This observation will be useful in constructing a likelihood function to estimate unknown parameters.

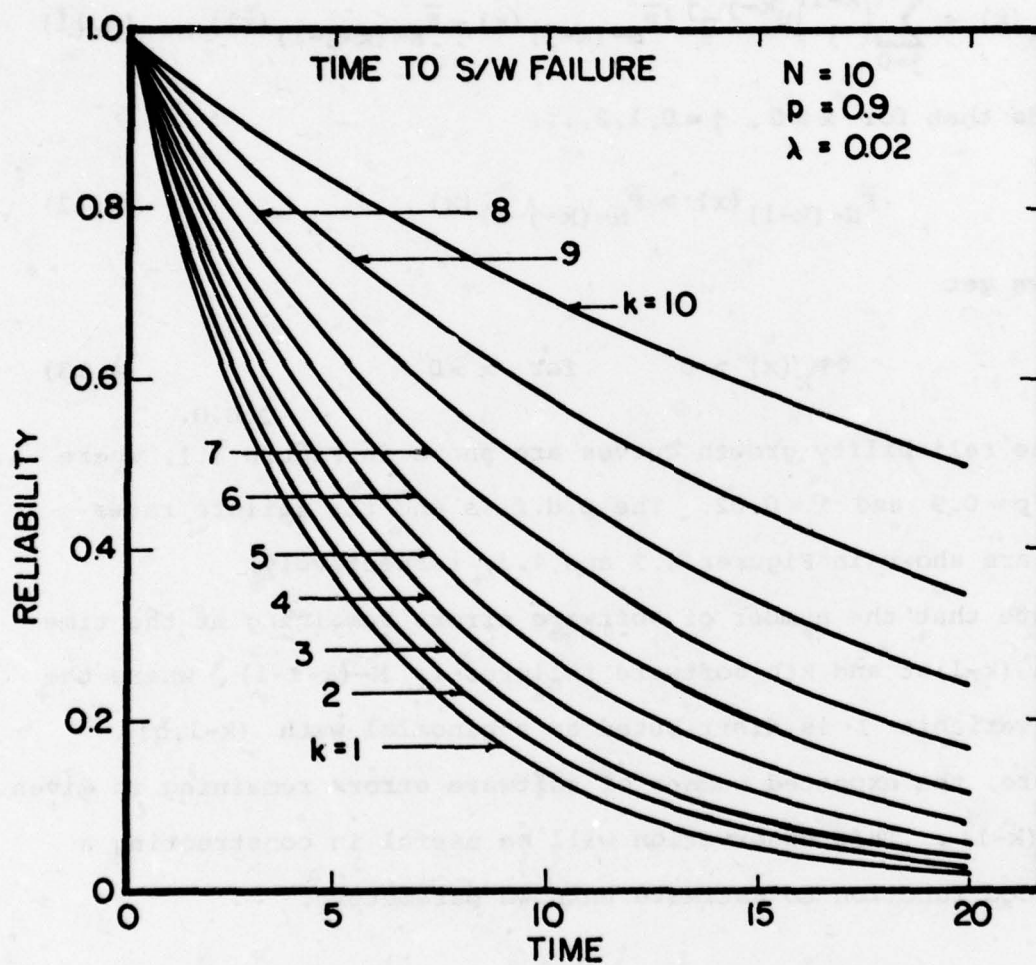


Figure 4.1 Reliability Growth Curves

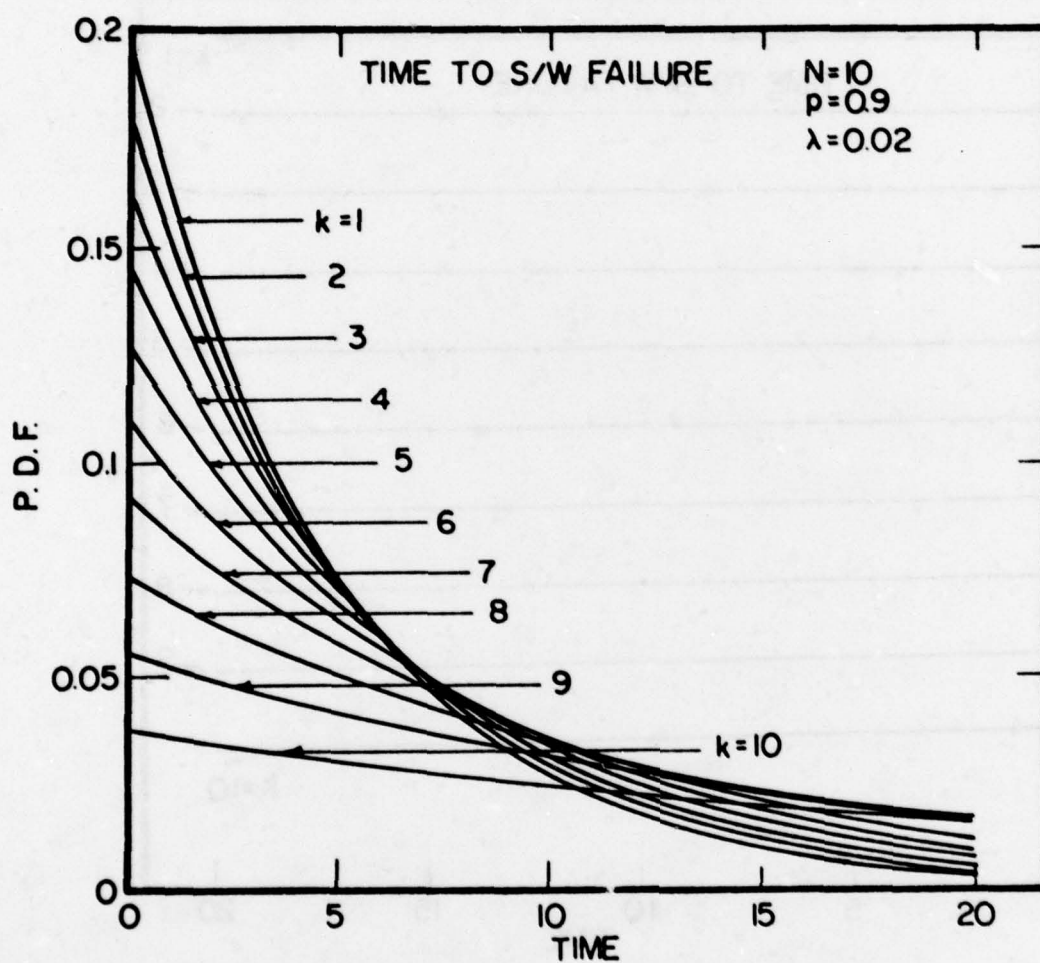


Figure 4.2 PDF of Time Between Software Failures

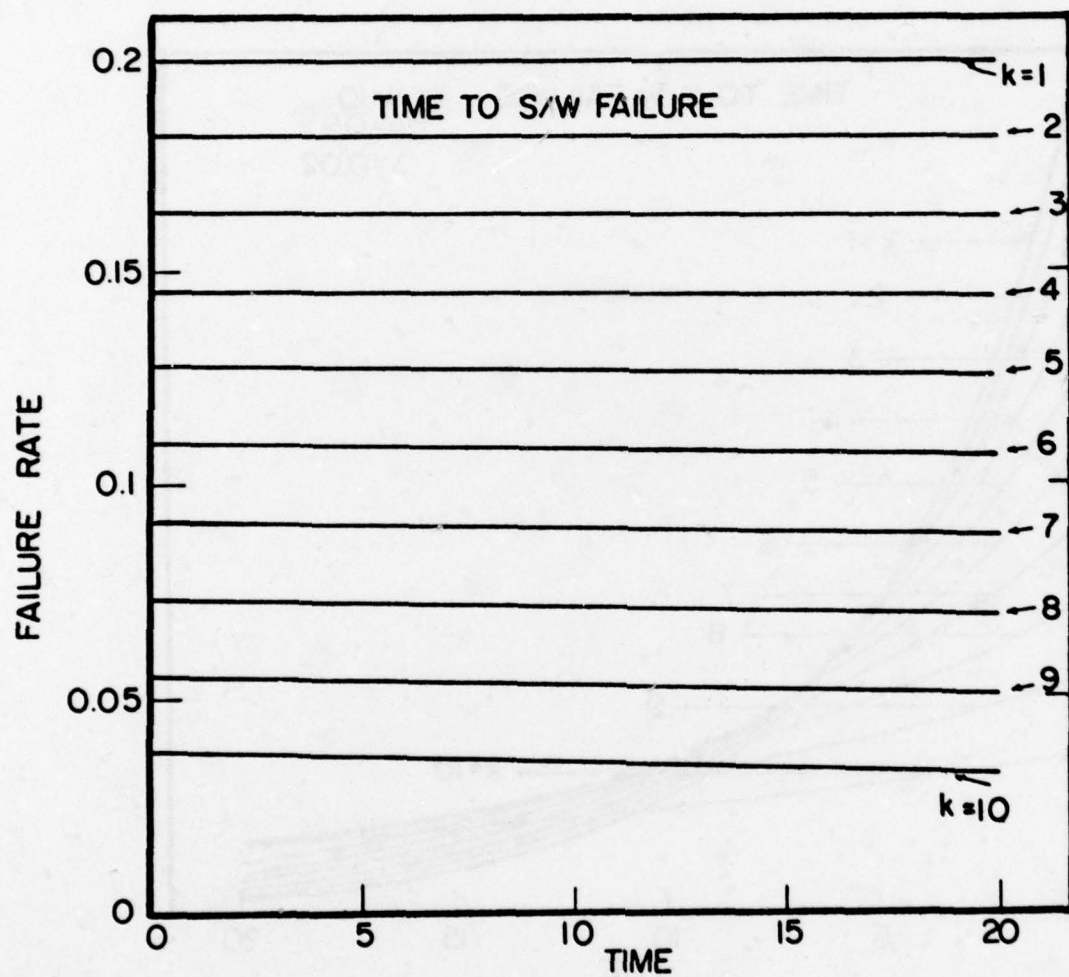


Figure 4.3 Failure Rates of Time Between Software Failures

5. GAMMA APPROXIMATION FOR A LARGE-SCALE SOFTWARE SYSTEM

In Section 3 we obtained the quantities of interest, e.g. state occupancy probability and renewal function, in terms of first passage time distribution. Once we have computed $G_{N,n_0}(t)$, we can easily obtain $P_{N,n_0}(t)$ and $M_N(t)$. However, it should be noted that the computation of $G_{N,n_0}(t)$ is almost impossible for a large-scale software system because of the difficulty in computing the coefficient, $B_{N,j,n}$. Through numerical study we have found that the computations become very messy and almost impossible for $N - n_0 \geq 20$. In this section we study methods for obtaining approximate solutions for these quantities.

Of prime interest is the approximation of first passage time distribution by using a Gamma distribution. From a study of the pdf's of first passage times in Figure 3.2, we feel that these distributions might be approximated by Gamma distributions. We use the method of moments to obtain estimates of the parameters of a Gamma distribution corresponding to $G_{N,n_0}(t)$. In order to do that, we first discuss how to obtain the moments of $G_{N,n_0}(t)$ without computing the coefficient B_{N,j,n_0} . Let T_{N,n_0} be a random variable which denotes the first passage time from N to n_0 . The random variable of holding time at state N , denoted by T_N , has an exponential distribution with parameter $N\lambda$. Therefore, we have

$$\mu_N = ET_N = 1/N\lambda \quad (5.1)$$

$$\text{Var}(T_N) = E(T_N - \mu_N)^2 = 1/(N\lambda)^2. \quad (5.2)$$

The following recursive equations are easily obtained:

$$\begin{aligned}
T_{N,n_0} &= T_N = pT_{N-1,n_0} + qT_{N,n_0} \\
T_{N-1,n_0} &= T_{N-1} + pT_{N-2,n_0} + qT_{N-1,n_0} \\
&\dots \\
T_{n_0+1,n_0} &= T_{n_0+1} + qT_{n_0+1,n_0}
\end{aligned}
\quad \left. \vphantom{\begin{aligned} T_{N,n_0} \\ T_{N-1,n_0} \\ \dots \\ T_{n_0+1,n_0} \end{aligned}} \right\} \quad (5.3)$$

Solving (5.3) recursively, we get

$$T_{N,n_0} = \frac{1}{p} \sum_{j=n_0+1}^N T_j. \quad (5.4)$$

Then, we have

$$\mu_{N,n_0} = ET_{N,n_0} = \frac{1}{p} \sum_{j=n_0+1}^N ET_j = \frac{1}{p} \sum_{j=n_0+1}^N 1/j\lambda \quad (5.5)$$

and

$$\text{Var}(T_{N,n_0}) = \frac{1}{p^2} \sum_{j=n_0+1}^N \text{Var}(T_j) = \frac{1}{p^2} \sum_{j=n_0+1}^N 1/(j\lambda)^2. \quad (5.6)$$

These are identical to the ones obtained in Section 3.2. Suppose the Gamma distribution corresponding to $G_{N,n_0}(t)$ has a shape parameter α and a scale parameter β , so the mean and variance are given by α/β and α/β^2 , respectively. Then the parameters α and β can be estimated by using the method of moments, i.e.,

$$\frac{1}{p} \sum_{j=n_0+1}^N 1/(j\lambda) = \alpha/\beta \quad (5.7)$$

$$\frac{1}{p^2} \sum_{j=n_0+1}^N 1/(j\lambda)^2 = \alpha/\beta^2. \quad (5.8)$$

Therefore, we have

$$\hat{\beta} = \frac{p \sum_{j=n_0+1}^N 1/(j\lambda)}{\sum_{j=n_0+1}^N 1/(j\lambda)^2} \quad (5.9)$$

$$\hat{\alpha} = \left[\sum_{j=n_0+1}^N 1/(j\lambda) \right]^2 / \sum_{j=n_0+1}^N 1/(j\lambda)^2. \quad (5.10)$$

Numerical examples for various n_0 are given in Table 5.1, where $N=100$, $p=0.9$ and $\lambda=0.02$. We also compute the relative losses for third and fourth moments around the mean to see how good the approximations are. Since the third and fourth moments around the mean of a Gamma distribution with parameters α and β are given by $2\alpha/\beta^3$ and $9\alpha/\beta^4$, respectively, we define the relative losses for third and fourth moments around the mean as

$$\frac{|E(T_{N,n_0} - \mu_{N,n_0})^3 - 2\alpha/\beta^3|}{E(T_{N,n_0} - \mu_{N,n_0})^3} \quad (5.11)$$

and

$$\frac{|E(T_{N,n_0} - \mu_{N,n_0})^4 - 9\alpha/\beta^4|}{E(T_{N,n_0} - \mu_{N,n_0})^4} \quad (5.12)$$

respectively, where

$$E(T_{N,n_0} - \mu_{N,n_0})^3 = \frac{1}{p^3} \sum_{j=n_0+1}^N E(T_j - \mu_j)^3 = \frac{2}{p^3} \sum_{j=n_0+1}^N 1/(j\lambda)^3 \quad (5.13)$$

and

Table 5.1
Gamma Approximation for First Passage Time Distributions
($N = 100$, $p = 0.9$, $\lambda = 0.02$)

n_o	Mean	Variance	$\hat{\alpha}$	$\hat{\beta}$	Relative Loss (%)	
					3rd Moment	4th Moment
10	125.47	263.01	59.85	0.477	28.15	4.21
15	103.84	168.34	64.05	0.617	21.58	2.70
20	88.31	119.82	65.09	0.737	16.81	1.92
25	76.19	90.31	64.28	0.844	13.19	1.44
30	66.24	70.47	62.27	0.940	10.37	1.12
35	57.81	56.23	59.44	1.028	8.14	0.89
40	50.49	45.49	56.04	1.110	6.35	0.72
45	44.02	37.12	52.21	1.186	4.92	0.58
50	38.23	30.40	48.07	1.257	3.77	0.47
55	32.99	24.90	43.70	1.324	2.84	0.39
60	28.19	20.30	39.15	1.389	2.09	0.31
70	19.70	13.07	29.69	1.507	1.03	0.20
80	12.33	7.63	19.92	1.616	0.41	0.11
90	5.82	3.39	9.99	1.716	0.09	0.05

$$\begin{aligned}
E(T_{N,n_0} - \mu_{N,n_0})^4 &= \frac{1}{p^4} \left[\sum_{j=n_0+1}^N E(T_j - \mu_j)^4 + 6 \sum_{i=n_0+1}^N \sum_{j>i} E(T_i - \mu_i)^2 E(T_j - \mu_j)^2 \right] \\
&= \frac{1}{p^4} \left[9 \sum_{j=n_0+1}^N 1/(j\lambda)^4 + 6 \sum_{i=n_0+1}^N \sum_{j>i} (1/i\lambda)^2 (1/j\lambda)^2 \right]. \quad (5.14)
\end{aligned}$$

Figure 5.1 shows the relative losses for third and fourth moments around the mean with N , where $p=0.9$, $\lambda=0.02$ and $n_0=0.2N$. As we see in this figure, the maximum relative losses for third and fourth moments around the mean are about 17% and 10%, respectively. This means that the Gamma approximation of first passage time distributions for large-scale software systems is reasonably good.

Plots of first passage time using Gamma approximation for $N=100$, $p=0.9$ and $\lambda=0.02$ are given in Figure 5.2 for $n_0=0,1,2,3,5$, and 9. Also, plots of state occupancy probabilities using this approximation are given in Figure 5.3.

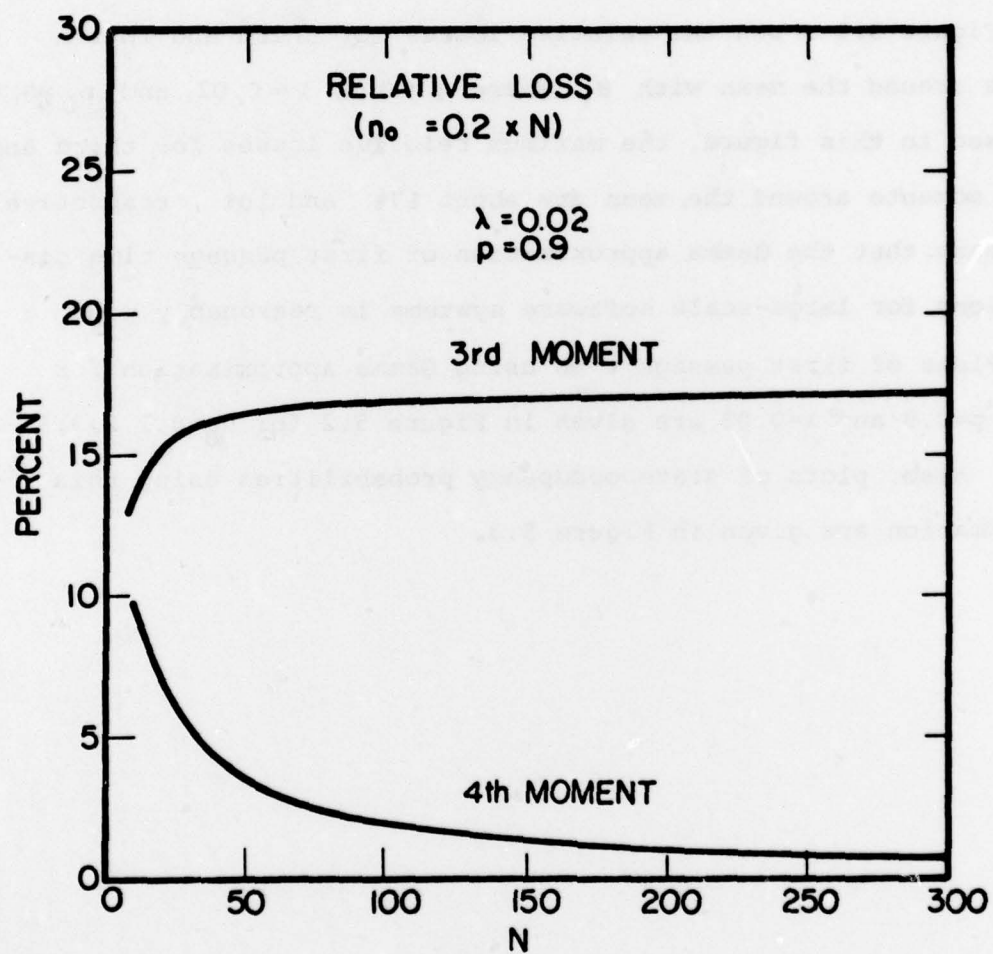


Figure 5.1 Relative Loss for the Third and Fourth Moments

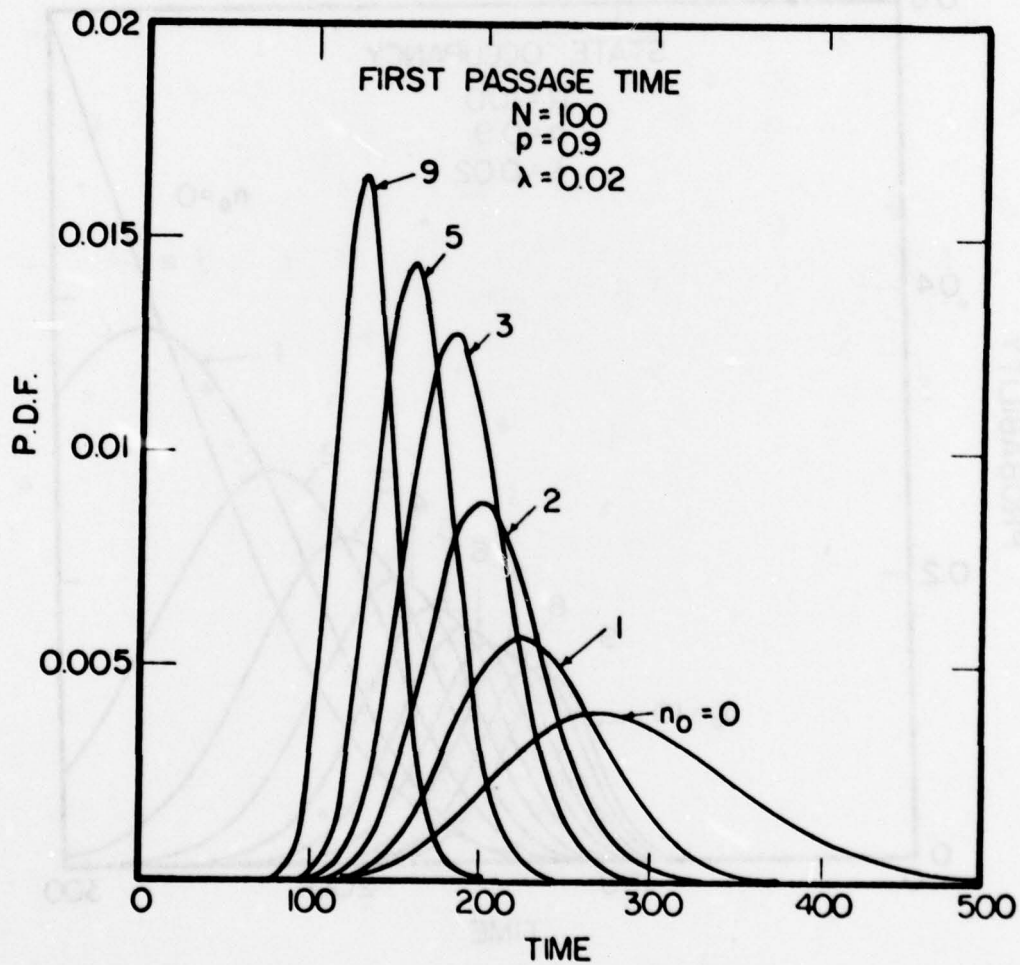


Figure 5.2 Plots of First Passage Times
Using Gamma Approximation

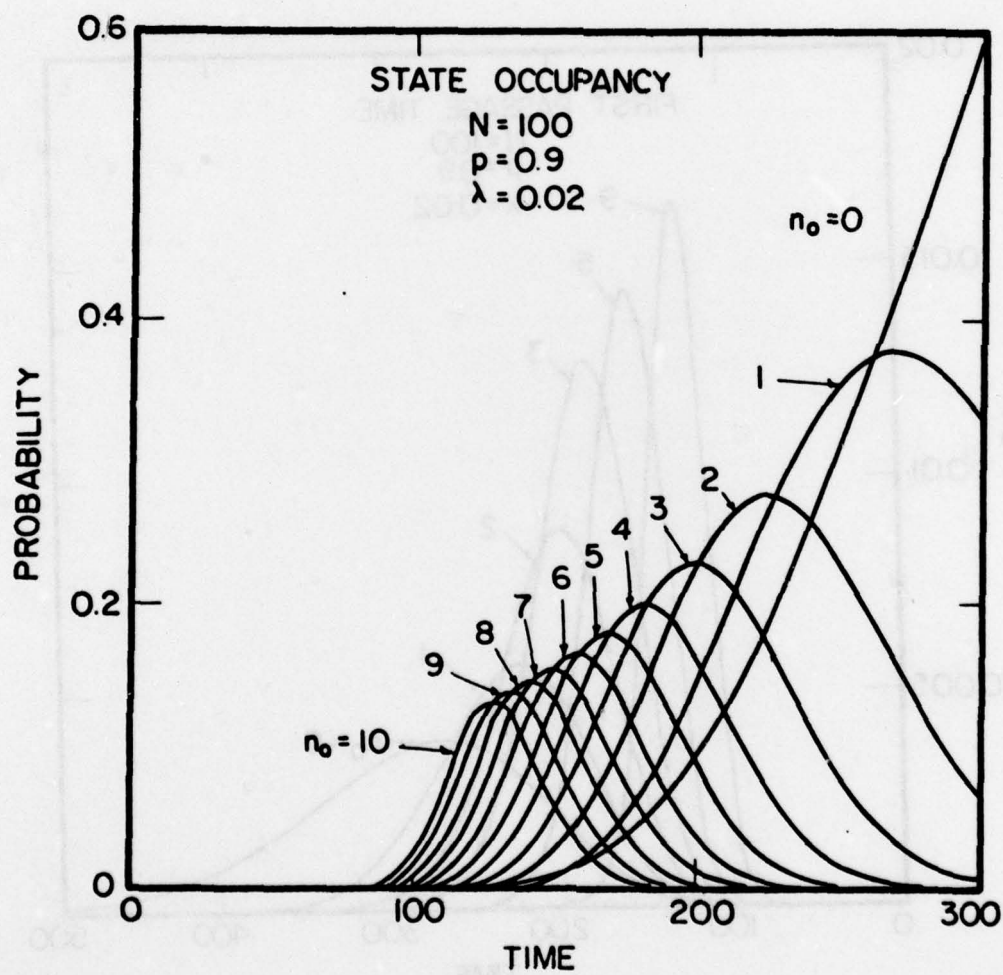


Figure 5.3 State Occupancy Probabilities
 Using Gamma Approximations

6. CONCLUDING REMARKS

An imperfect debugging model (IDM) for software systems was developed in this report. Various quantities of interest were derived in terms of the first passage time distribution of the underlying semi-Markov process. Computations for and usefulness of these quantities were illustrated via numerical examples. An approximation method for obtaining these quantities for large-scale software system was also presented.

It should be pointed out that most of the models reported in the literature, for example the models in [3], [6], [9], [10], and [13], are special cases of IDM.

7. SELECTED REFERENCES

- [1] Barlow, R. E. and Proschan, F. (1965), Mathematical Theory of Reliability, Wiley.
- [2] Endres, A., (1975), "An Analysis of Errors and Their Causes in System Programs," Proceedings: 1975 International Conference on Reliable Software, pp. 327-336.
- [3] Jelinski, J. and Moranda, P. B. (1972), "Software reliability research," in Statistical Computer Performance Evaluation, edited by W. Freiberger, Academic Press.
- [4] Jelinski, J. and Moranda, P. B. (1973), "Applications of a probability-based model to a code reading experiment," Record: 1973 IEEE Symposium on Computer Software Reliability, pp. 78-81.
- [5] Littlewood, B. (1975), "A Reliability Model for Markov Structured Software," Applied Statist., Vol. 24, pp. 172-177.
- [6] Littlewood, B. and Verrall, J. L. (1973), "A Bayesian reliability growth model for computer software," Appl. Statist., Vol. 22, pp. 332-346.
- [7] Miyamoto, I. (1975), "Software Reliability in On-line Real Time Environment," Proceedings: 1975 International Conference on Reliable Software, pp. 194-203.
- [8] Moranda, P. (1975), "A comparison of software error-rate models," 1975 Texas Conference on Computing.
- [9] Musa, J. D. (1975), "A Theory of Software Reliability and Its Application," IEEE Trans. on Software Engineering, Vol. SE-1, No. 3, pp. 312-327.
- [10] Schick, G. J. and Wolverton, R. W. (1972), "Assessment of Software Reliability," McDonnell-Douglas Astronautics Company Paper WD1872.
- [11] Schneidewind, N. (1972), "An Approach to Software Reliability Prediction and Quality Control," AFIPS Conference Proceedings, Vol. 41, Part II, Fall Joint Computer Conference, pp. 837-838.

- [12] Schneidewind, N. J. (1975), "Analysis of Error Processes in Computer Software," Proceedings: 1975 International Conference on Reliable Software, pp. 337-346.
- [13] Shooman, M. L. (1972), "Probabilistic Models for Software Reliability Prediction," Statistical Computer Performance Evaluation, pp. 485-502, Academic Press, New York.
- [14] Shooman, M. L. and Bolsky, M. I. (1975), "Types, Distribution, and Test Correction Times for Programming Errors," Proceedings: 1975 International Conference on Reliable Software, pp. 347-357.
- [15] Thayer, T. A. et al (1976), "Software Reliability Study," TRW Defense & Space Systems Group, Final Technical Report RADC-TR-76-238, August 1976. (A030798)
- [16] Trivedi, A. K. and Shooman, M. L. (1975), "A Many-State Markov Model for the Estimation and Prediction of Computer Software Performance Parameters," Proceedings: 1975 International Conference on Reliable Software, pp. 208-220.
- [17] Wagoner, W. L. (1973), "The Final Report on Software Reliability Measurement Study," Aerospace Report No. TOR-0074(4112)-1.
- [18] Wolverton, R. W. (1974), "The cost of developing large scale software," IEEE Trans. on Computers, vol. C-23, No. 6, pp. 615-636.